École d'Ingénieurs du
Littoral-Côte-d'Opale

TP : Réseaux industriels et supervision

Interface Pygame

Auteur: *Fono Colince*

Supervisé par:
Mr. Pierre Chatelain

Date:
May 19, 2024

Contents

1 Introduction

Dans ce TP, vous allez utiliser un simulateur d'automate qui simulera un automatisme industriel. Vous allez ensuite communiquer avec cet automate en utilisant le protocole Modbus pour produire une supervision.

2 interface personnelle

3 Pendant le cycle de remplissage

4 Pendant le cycle de vidange

5 Annexe

5.1 La liste des icônes créé

5.2 Un diagramme des classes

5.3 Vos codes sources

```
import pygame
import time
from math import pi, cos, sin
import random
from modbus import Modbus

import sys

# Initialisation de Pygame et de l' cran
pygame.init()
modbus = Modbus()
ecran = pygame.display.set_mode((800, 600))
pygame.display.set_caption("Simulation de R servoir de Liquide")
clock = pygame.time.Clock()

# Couleurs
GREEN = (0, 255, 0)
BLACK = (0, 0, 0)
RED = (255, 0, 0)
BLUE = (0,0, 255)
LIQUIDE = (random.randint(1, 254),random.randint(1, 254), 255)
GREY = (169, 169, 169)
AMPOULE = [pygame.image.load('./Assetes/Aon.png'),pygame.image.
    load('./Assetes/Aoff.png')]


# G n ration initiale des bulles
def generate_bubbles(num_bubbles):
    return [(random.randint(10, 90), random.randint(10, 90),
    random.randint(2, 5)) for _ in range(num_bubbles)]
```

```
29
30 # G n ration initiale des water
31
32 # Initialisation des bulles
33 bubbles = generate_bubbles(15)
34
35 # Animation des bulles
36 def animate_bubbles(bubbles, height):
37     for i in range(len(bubbles)):
38         bubbles[i] = (bubbles[i][0], (bubbles[i][1] - 0.5) % 100,
    bubbles[i][2])
39
40
41 # Fonction pour dessiner un indicateur de niveau de liquide
42 def draw_level_indicator(surface, x, y, fill_level):
43     font = pygame.font.Font(None, 36)
44     level_text = font.render(f"{fill_level:.1f}%", True, BLACK)
45     surface.blit(level_text, (x, y))
46
47 # Fonction pour dessiner le r servoir
48 def draw_tank(surface, x, y, width, height, fill_level):
49
50     # Dessin du liquide
51     liquid_height = fill_level / 100 * height
52     pygame.draw.rect(surface, LIQUIDE, (x + 2, y + height -
    liquid_height, width, liquid_height),1)
53
54
55     # Ajout d'un effet de d grad  pour le liquide
56     for i in range(1, int(liquid_height), 2):
57         alpha = 255 - int(255 * (i / liquid_height))
58         s = pygame.Surface((width, 2), pygame.SRCALPHA)
59         s.fill((LIQUIDE[0],LIQUIDE[1], 255, alpha))
60         surface.blit(s, (x + 2, y + height - i))
61
62     #tank
63     pygame.draw.line(surface, BLACK, (x, y), (x, y+height), 3)
64     pygame.draw.line(surface, BLACK, (x, y+height), (x+width, y+
    height), 3)
65     pygame.draw.line(surface, BLACK, (x+width, y+height), (x+
    width, y), 3)
66
67     # Dessiner et animer les bulles
68     draw_bubbles(surface, x, y, width, height, fill_level,
    bubbles)
69     animate_bubbles(bubbles, 10)
70
71     #
    ----------------------------------------------------------------------------------------------------
72
73 def draw_bateri(surface, x, y, width, height, fill_level):
74     # Dessin du liquide
75     col = LIQUIDE
76     liquid_width = fill_level / 100 * width
77     if liquid_width < width/2:
78         col = RED
```

```
    else:
        col = LIQUIDE

    pygame.draw.rect(surface,col,(x, y , liquid_width, height))
    pygame.draw.rect(surface, BLACK, (x, y , width, height),2)

    draw_level_indicator(surface,x+width,y,fill_level)


def draw_indicator(surface,x,y,l1,l2,l3):
    draw_bateri(surface, x, y, 400, 20, l1)
    draw_bateri(surface, x, y+30, 400, 20, l2)
    draw_bateri(surface, x, y+60, 400, 20, l3)




# Fonction pour dessiner des cercles (bubbles) dans le liquide
def draw_bubbles(surface, x, y, width, height, fill_level,
    bubbles):
    liquid_height = fill_level / 100 * height
    for bubble in bubbles:
        bubble_y = y + height - bubble[1] * liquid_height / 100
        pygame.draw.circle(surface, BLUE, (x + bubble[0] * width
    / 100, int(bubble_y)), bubble[2])



def rotate_image(ecran,x,y,image, angle):
    rotated_image = pygame.transform.rotate(image,angle)
    rot_img_rect = rotated_image.get_rect(center=(x,y))
    ecran.blit(rotated_image,rot_img_rect)

def tapie(ecran,x,y,angle,img1,dir,stat,ampoule):
    ampouleRect = ampoule[stat].get_rect()
    ampouleRect.x = x-60
    ampouleRect.y = y-40
    if stat==0:
        ecran.blit(ampoule[stat],ampouleRect)
    else:
        angle=0
        ecran.blit(ampoule[stat],ampouleRect)

    rotate_image(ecran,x,y,img1,angle*dir)
    rotate_image(ecran,x+100,y,img1,angle*dir)

    ecart = (angle/360)*50
    pygame.draw.line(ecran, BLACK, (x+ecart*(dir), y-15), (x+
    ecart*(dir)+10, y-15), 3)
    pygame.draw.line(ecran, BLACK, (x+ecart*(dir)+50, y-15), (x+
    ecart*(dir)+10+50, y-15), 3)

    pygame.draw.line(ecran, BLACK, (x+ecart*(-dir)+90, y+15), (x+
    ecart*(-dir)+100, y+15), 3)
    pygame.draw.line(ecran, BLACK, (x+ecart*(-dir)+50, y+15), (x+
    ecart*(-dir)+10+50, y+15), 3)

```

```
130
131
132 def main():
133
134     continuer = True
135
136     # asset
137     roue = pygame.image.load('Assetes/roue.png')
138     angle = 0.0
139     vetesse_roue = 3.0
140
141
142     clock = pygame.time.Clock()
143
144     while continuer:
145         for event in pygame.event.get():
146             if event.type == pygame.QUIT:
147                 continuer = False
148             elif event.type == pygame.KEYDOWN:
149                 if event.key == pygame.K_q:
150                     continuer = False
151                 elif event.key == pygame.K_UP:
152                     print("K_UP")
153                     #fill_level = min(100.0, fill_level + 1.0)
154                 elif event.key == pygame.K_DOWN:
155                     print("K_DOWN")
156                     #fill_level = max(0.0, fill_level - 1.0)
157
158         ecran.fill((224, 224, 224))  # Couleur de fond
159         fill_level = modbus.lireRegistre(30)# niv + vitesse.y
160         print(fill_level)
161
162         #conv
163         SUP_CONV1 = modbus.lireBit(303)
164         SUP_CONV2 = modbus.lireBit(304)
165         SUP_CONV3 = modbus.lireBit(305)
166
167         #menu
168         draw_indicator(ecran,200,50,100,100,fill_level/100)
169
170         # Dessiner le r servoir avec le niveau de liquide
171         draw_tank(ecran, 200, 200, 100, 100, 100)
172         draw_tank(ecran, 500, 200, 100, 100, 100)
173         draw_tank(ecran, 350, 400, 100, 100, fill_level/100)
174
175
176         angle += vetesse_roue
177         if angle >=360:
178             angle =0
179
180         if SUP_CONV3==1:
181             SUP_CONV3=0
182         else:
183             SUP_CONV3=1
184
185         if SUP_CONV2==1:
186             SUP_CONV2=0
```

```
        else:
            SUP_CONV2=1

        if SUP_CONV1==1:
            SUP_CONV1=0
        else:
            SUP_CONV1=1


        tapie(ecran,200,350,angle,roue,1,SUP_CONV1,AMPOULE)
        tapie(ecran,500,350,angle,roue,1,SUP_CONV2,AMPOULE)
        tapie(ecran,350,550,angle,roue,1,SUP_CONV3,AMPOULE)
        clock.tick(60)




        pygame.display.flip()
        clock.tick(30)

    pygame.quit()

if __name__ == "__main__":
    main()
```

Code source 1: .